NPS52-83-009

NAVAL POSTGRADUATE SCHOOL

Monterey, California

COMPUTER SYSTEM DESIGN ENVIRONMENT
SOFTWARE DEVELOPMENT PLAN

Ms. Jeanne Bowers
Lt.Col Alan A. Ross

July 1983

Chief of Naval Research
Arlington, VA 22217

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral J. J. Ekelund
Superintendent

D. A. Schrady
Provost

The work reported herein was supported in part by the Foundation Research Program of the Naval Postgraduate School with funds provided by the Chief of Naval Research.

This report was prepared by:

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NPS52-83-009	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE *(and Subtitle)*: Computer System Design Environment Software Development Plan	5. TYPE OF REPORT & PERIOD COVERED: Technical Report
6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(s): Professor Alan A. Ross, Jeanne Bowers	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School, Monterey, California 93943	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61152N; RR000-01-100, N0001483WR30104
11. CONTROLLING OFFICE NAME AND ADDRESS: Chief of Naval Research, Arlington, VA 22217	12. REPORT DATE: July 1983
	13. NUMBER OF PAGES: 21
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*	15. SECURITY CLASS. *(of this report)*: Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

primitives	computer aided design	interrupt-driven monitor
user-friendly	structured development	syntax directed editor
data base	translator	design description language
prototype	mapper	

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The Computer Systems Design Environment (CSDE) project is an attempt at automated design of computer systems. The project develops a system which will accept functional statements of requirements from the designer (utilizing a user-friendly dialogue); translate those requirements into software and hardware primitives; evaluate those primitives and develop a proposed system using a Library of Realization Volumes. The CSDE will also verify that timing requirements have been met by the proposed hardware and software; and present

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

the systems design to the designer (in a user-friendly format). The CSDE will be a prototype based on an existing feasibility demonstration version which has verified the concept. The prototype version will explore issues of adaptability, user friendliness and design system performance. This paper is a plan for the development of the CSDE.

COMPUTER SYSTEM DESIGN ENVIRONMENT
SOFTWARE DEVELOPMENT PLAN
July 29, 1983

I. Purpose of Plan

The purpose of this Software Development Plan is to describe the development approach for the Computer Systems Design Environment (CSDE), to schedule the development and implementation, and to provide guidance to the researchers, staff and students involved in the project. The document will be used primarily in-house and, therefore, it is written to provide the project personnel with a direction for the overall effort, guidelines for their individual tasks, and definition of how their thesis topic relates to the entire project. The document will require modification as the project progresses, and will be updated once a quarter to reflect any required changes in direction or schedule.

II. Background

The basic concept for the CSDE was inspired by previous research conducted by M. N. Matelan on the design of real time control systems. This CSDE concept has been developed by Professor Alan Ross and Professor Herschel Loomis at the Naval Postgraduate School. Currently, there is a functioning model which was developed as a feasibility demonstration and has confirmed that the concept is viable. The current model is cumbersome to use and only provides the most basic capabilities. However, it does generate a system design for a set of functional requirements based on a library of primitives.

III. Scope

The Computer System Design Environment is a set of tools for the computer system designer. The CSDE will allow the designer to input the requirements of a problem in functional terms. The CSDE will examine the requirements and will automatically generate a digital-processor based solution to the problem. The design tools will apply the concepts of very high level languages, syntax directed editors, designer's workstations, scheduling theory, and hardware and software descriptive languages to build libraries of problem solution modules, and then to select among these modules to design a particular system.

The Design Environment is modularized and, as shown in Figure 1, there are five basic modules: the Input Translator, the Functional Mapper, the Timing Analyzer, the Library

Updater, and the Output Generator. The user of the system interfaces at three points with the modules: he inputs problem descriptions into the Input Translator; he inputs device descriptions into the Library Updater; and he receives the results on a display/graphics terminal or printed report. The Library Updater module is an independent process which provides the Library of Realization Volumes to the Functional Mapper and Timing Analyzer. The four other modules (excluding Library Updater) have a process and data flow from one to another as illustrated in Figure 1.

The objectives of the CSDE project are to develop a prototype version of the system using applicable software engineering techniques and to enhance the functional capabilities of the feasibility demonstration version with improvements and modifications developed while testing and using it. The result of this effort will not be a production system; if the prototype is successful, a production version will be developed and implemented. The system developed will generate computer system designs for embedded digital computers within weapon systems. An attempt will be made, as part of the project, to expand the concept to other classes of problems. The goal is to develop a single design system which will satisfy a maximum number of problems. The CSDE system is a prototype and although accepted software engineering techniques will be used, the documentation will be less complete than is required for a production system. Guidelines for the preparation of documentation are included in Appendix A.

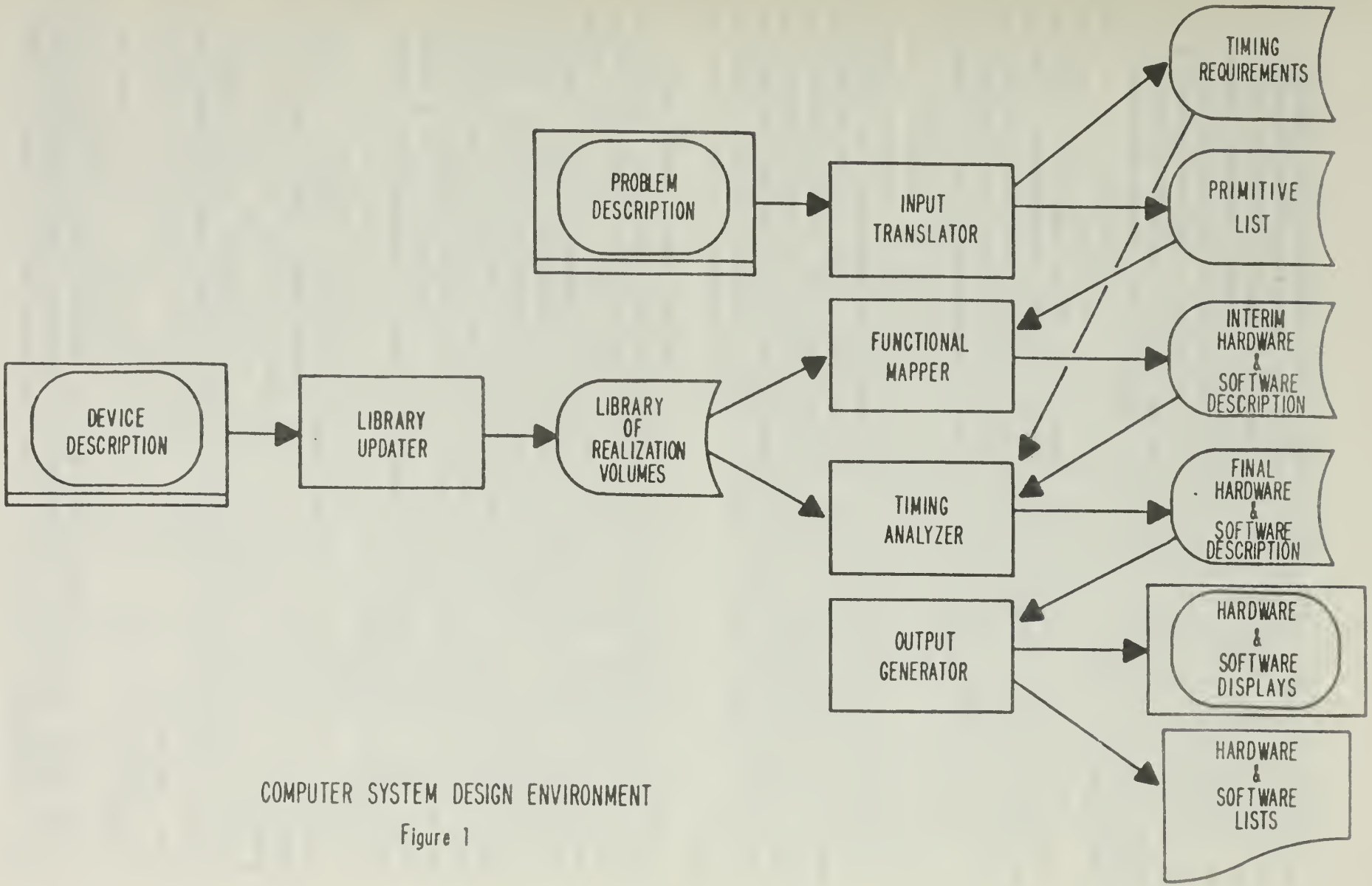
IV. General Approach to Development

The systems development approach to be used in this effort is based on a systems design methodology, structured programming, and the Oracle data base development tools. Structured walkthroughs will be conducted by the thesis student for the project personnel throughout the development cycle. All documentation will be done using an automated text editor. The following sections will describe the general development steps, the hardware and software environment, and the relationships among the participants of the project.

A. Development Steps

The following paragraphs describe the general steps to be used for developing software for the project:

- Plan development - This step includes a detailed analysis of the tasks that need to be done and the time frames



COMPUTER SYSTEM DESIGN ENVIRONMENT

Figure 1

required based on the level of participation. During this step, a Milestone Chart will be developed which will be revised as the project evolves. These milestones will be established using the Development Steps listed below as a guideline. The time frame estimate in this Software Development Plan for the overall task or subtask should also be considered. If necessary, the individual milestone charts will dictate a change to this Software Development Plan.

- Define/verify detailed requirements - This step includes a detailed definition of what the specific module of the CSDE should be capable of doing. The process to be used to define these requirements consists of the following: a functional analysis of the module including a decomposition and a concise description for each subfunction; a definition of the data categories required to support the function; a description of the relationship between the subfunctions and the data; an identification of the transactions required to drive the module; and a structure for the data base including the required categories and their relationships. During this step, the Requirements Document will be produced (See Appendix A for guidelines on the contents and structure of the Requirements Document).

- Evaluate existing module to determine task approach - This step includes analyzing the existing program module and determining how to proceed. The existing module must be functionally analyzed to determine how many of the required functions are currently included in the feasibility version. The data currently generated and/or used by the existing module must be examined and compared with what is required of the module which is being developed. The results of this step will be used to determine which succeeding steps will be done and to what degree they must be done. A synopsis of the findings of this step and the conclusions concerning the subsequent steps should be prepared.

- Design new module or enhancement to current module - This step includes the design of the detailed logic of the programs or modifications necessary to meet the requirements previously defined. The output from this step will be program specifications and/or modification specifications (See Appendix A for guidelines in preparing program and program modification specifications). All logic developed for the CSDE will use a structured approach.

- Code and test new module or enhancement - This step includes the actual programming, testing and debugging of the computer software. The language which will be used

for programming will be determined by the primary researcher; the language will be compatible with the Oracle Data Base Management System. Programming will use structured techniques and development and testing of programs will be, as much as practical, top-down. Testing will be done at a program level first and then at a combined level with the other programs with which the program interfaces. An informal test plan will be developed before testing begins. Test data and results will be retained for documentation. The programs should be well commented and readable using good indentation techniques. The documentation from this step will be an updated and amplified (where applicable) specification with the attachment of appropriate test data and results (See Appendix A for guidelines in producing maintenance documentation).

- Document new module or enhancement - Since we have previously discussed documentation at the specification and maintenance level, this step includes only user documentation. User "help" screens are strongly encouraged, but a small user document is required to supplement the online documentation for procedures such as initially getting on the system (See Appendix A for guidelines in producing user documentation).

- Perform final testing and integration into the system - This step includes the interfacing of the new module with the currently existing CSDE. The interfacing requirements should be verified to ensure that nothing has changed since the original requirements were identified. An outline of the steps necessary to integrate the module should be prepared and scheduled with the primary researcher. This step is a critical one and one which can very easily cause serious problems. Coordination, communication and planning are the key to success.

If, during the course of any of these development steps, the thesis student determines either the need for splitting his topic or for adding an additional topic, he should inform his thesis advisor who should forward the information to the primary researcher immediately.

B. Environment - The following paragraphs describe the hardware and software environments in which the CSDE will be developed.

1. Hardware - The development work will be accomplished using a Digital Equipment Corporation VAX 11-780. VAX Workstations or equivalents will be procured to act as designer workstations. As many high-level tools as possible will be implemented for the designer at the workstation to

make his function easier.

2. Software - The Digital Equipment Corporation VAX VMS Operating System will be used as the operating software environment. In addition, the Oracle Data Base Management System and its design aids will be used where applicable. The EDT editor will be used for documentation and program editing. The specific language compiler/interpreter will be determined by the primary researcher.

C. Staffing - The following paragraphs describe the responsibilities of each category of the project staff. Each thesis study will be conducted by a thesis student with guidance from a thesis advisor and assistance from the Computer Science Department staff.

1. Researchers - The primary researcher is LTCOL Alan Ross; he will be assisted by associate researchers Professor Herschel Loomis, Professor George Rahe, and Captain Bradford Mercer. The responsibility of the primary researcher is one of planning, control and coordination. He must keep the overall project progressing and must assure that the pieces fit together at the appropriate times. He will administer the project but will do so from a very technical point of view. The reports which must be produced for sponsors of the research will be prepared by the primary researcher with help from his associates. Each associate researcher has one or more area(s) of specific interest and will be involved in the researching and development of those areas. The specific areas of interest for the associate researchers are: Professor Loomis, timing analysis and the implementation of the Realization Library; Professor Rahe, man-machine interface; and Captain Mercer, representations and methods for describing systems and primitives. All researchers will act as thesis student advisors.

2. Students - The students associated with the project will be thesis students developing one aspect of the CSDE as their thesis topic. Due to the significant turnover in the students participating in the project, a project staff person will be peripherally involved in all thesis work. An official turnover of programs and supporting documentation to the project staff person will be required of each student prior to his or her departure. The students can request assistance from the project staff and the Computer Science Department staff, as required.

3. Staff Support - The support available to the project includes both project and Computer Science Department Staff. The project staff includes one full-time Programmer/Analyst, one part-time Technician/Data Entry, and a small amount of clerical support. The Computer Science Department staff

members will not be dedicated to the project, but will be available to assist the project staff, students and researchers, as required, specifically in areas such as the use of the development methodology, VMS and ORACLE.

V. Assumptions

The following is a list of items which are assumed to be true for the development process.

- Funding will be available for staffing in accordance with the Computer System Design Environment Project Proposal dated May 23,1983.
- The Oracle Data Base Management System will be available for use by the project.
- Support will be available from the Computer Science Department Staff in the areas of the system design methodology being used, VMS and ORACLE.

VI. Detailed Plan

Section III identified five modules which make up the CSDE. These five modules and two additional areas are now addressed in greater detail.

A. Functional Breakdown

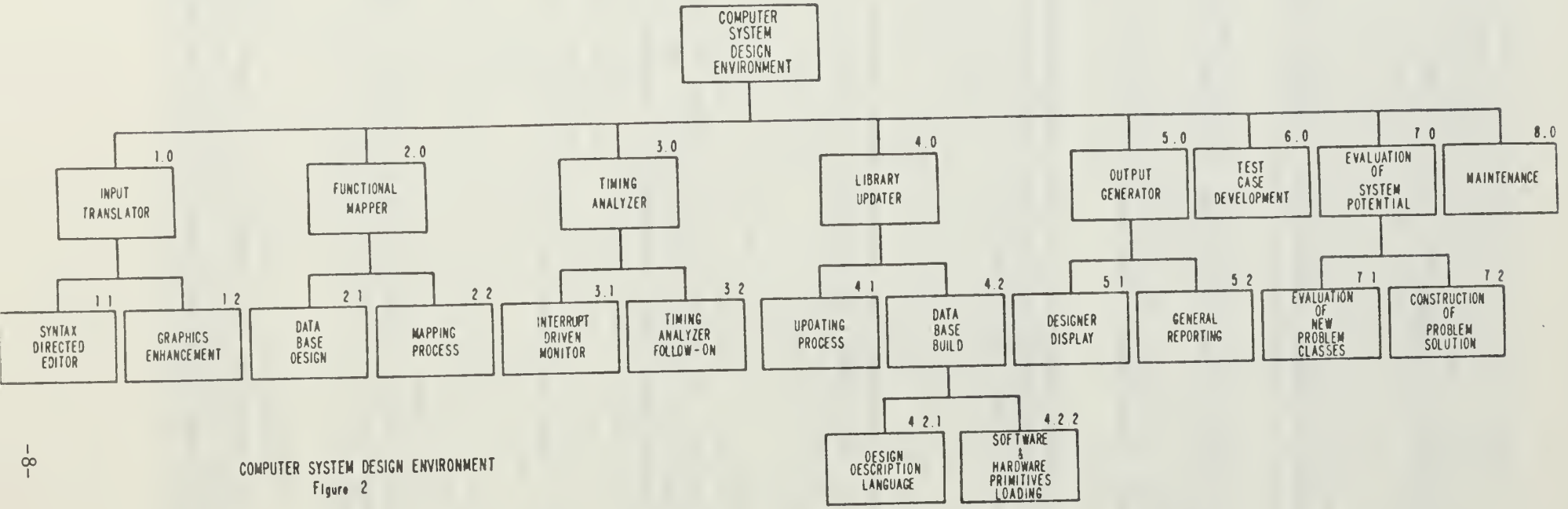
The process of systems development will include decomposing the system into its smallest functional pieces. In order to adequately plan the development and identify thesis topics, the initial steps of this decomposition must be accomplished. Figure 2 illustrates the results of this process down to the specific tasks and subtasks. Additional decomposition will be done during the analysis phase of the development.

B. Description of Functions

The following paragraphs will describe the tasks and subtasks of the project and will indicate which of these will be separate thesis topics.

1. Task 1 - Input Translator

The major goal of this task is to produce a user-friendly environment which will allow the designer to input the requirements of his system using a functional approach and



COMPUTER SYSTEM DESIGN ENVIRONMENT
Figure 2

an easily understandable language. The task has been divided into two subtasks with each representing a potential thesis topic.

a. Subtask 1.1 - Syntax Directed Editor

This subtask includes the overall analysis of the requirements of the Input Translator and the completion of the development of a syntax directed editor. The current thesis work of LCDR Barbara Sherlock will be the basis for this subtask.

b. Subtask 1.2 - Graphics Enhancement

This subtask includes the analysis and evaluation of the use of graphical techniques for inputting system requirements. The subtask determines if graphics can be feasibly interfaced, and, if so, accomplishes the interface.

2. Task 2 - Functional Mapper

The major goal of this task is to produce an environment for a simpler and more efficient mapping of requirements to available components. The task has been divided into two subtasks with each representing a potential thesis topic.

a. Subtask 2.1 - Data Base Design

This subtask includes the overall design of the entire data base for the CSDE. This data base design will include both a logical and physical design (based on Oracle as the data base management system). The approach will be to do the entire design as a separate subtask and thesis project rather than to split the data base design according to the functions that the data support. This approach will provide consistency of design and coordination of commonly used data. The design will be susceptible to modification as detailed development of other tasks is undertaken. Since the ability to respond easily to requirements modification is a strength of data base management systems, this approach was selected. Once this subtask is complete, it is important that a Data Administration function be established to assure continuity, consistency and coordination.

b. Subtask 2.2 - Mapping Process

This subtask includes the overall analysis of the requirements of the functional mapping process and the satisfaction of those requirements in a data base environment using the Oracle Data Base Management System. The data base design will not be a part of this topic, but the implementation of that design for the tables used by the Mapping Process and not previously implemented by another task will be included.

3. Task 3 - Timing Analyzer

The major goal of this task is to produce a monitor which will efficiently analyze and evaluate the timing requirements of the problem, and compare them with the speed of the system generated in the Functional Mapper. The task has been divided into two subtasks with each representing a potential thesis topic.

a. Subtask 3.1 - Interrupt-Driven Monitor

This subtask includes the overall analysis of the requirements of the Timing Analyzer and the development of an Interrupt-Driven Monitor strategy which will satisfy those requirements.

b. Subtask 3.2 - Timing Analyzer Follow-on

This subtask is not clearly defined at this point. It is listed because it is believed that following the interrupt-driven approach to the monitor, further work will be required to refine the approach or develop a new one.

4. Task 4 - Library Updater

The major goal of this task is to achieve a Library of Realizations with both hardware and software primitives loaded into the database using hardware and software design languages. The task has been divided into two subtasks. The first subtask represents a potential thesis topic; the second breaks down further into two additional subtasks with each representing a potential thesis topic.

a. Subtask 4.1 - Updating Process

This subtask includes the implementation of the library portion of the previously designed data base and the development of the software to allow adding, changing and deleting realizations.

b. Subtask 4.2 - Data Base Build

This subtask includes the loading of hardware and software primitives into the data base. It has been further divided into two subtasks with each representing a potential thesis project.

(1) Subtask 4.2.1 - Design Description Language

This subtask includes the analysis and evaluation of currently existing design description languages and the selection of one of these languages for use or the development of a new one. This language should address both

hardware and software aspects.

(2) Subtask 4.2.2 - Software and Hardware Primitive Loading

This subtask utilizes the language developed or selected in Subtask 4.2.1 and includes the analysis and research to determine the software and hardware primitives, the coding and data entry of these primitives and the conversion of any existing data to conform to the new data base.

5. Task 5 - Output Generator

The major goal of this task is to give the designer the most readable, useable display of the results of the design process possible. Although this task was divided into two subtasks, only the first will provide a potential thesis topic. The second subtask will be accomplished as a part of all other development tasks.

a. Subtask 5.1 - Designer Display

This subtask includes the analysis and development of the optimum display of the design results for the designer. The analysis should consider all aspects of usage of the information and graphics should be considered as an alternative approach. This subtask will be strongly tied to Task 1.

b. Subtask 5.2 - General Reporting

This subtask will be split among the other development tasks. It includes the development of all general reports which are required for each development area. This report generation is normally a part of the applications development process, but it is mentioned to distinguish it from the Designer Display subtask.

6. Task 6 - Test Case Development

This task includes producing the capability to prove, verify and validate a problem solution. The task will require developing testing techniques and data to prove the workability of problem solutions. It will have to be addressed in several phases until the solution to any problem which can be stated to the Input Translator and resolved by the Functional Mapper and Timing Analyzer can be verified by the test scenarios. The initial areas of investigation might be producing a check list for the technician, (i.e., if we force a given input, we can expect a certain output); and, developing a technique to analyze, evaluate and verify functionality and timing.

7. Task 7 - Evaluation of System Potential

The major goal of this task is to determine if different classes of problems can be satisfied with the existing (at that time) system. The task has been divided into two subtasks with each representing potentially several thesis topics.

a. Subtask 7.1 - Evaluation of New Problem Classes

This subtask includes applying a new class of problem to the CSDE and determining if the new class can be satisfied with the existing system or whether modifications are necessary. If modifications are required, they must be developed and implemented as a part of this task. Each new class of problem constitutes a separate thesis topic.

b. Subtask 7.2 - Construction of Problem Solution

This subtask includes the actual physical construction of the hardware and software dictated by the solution, and the true testing of the system, i.e., to actually perform the functions described by the designer in his original description of the problem using the recommended machine and software.

8. Task 8 - Maintenance

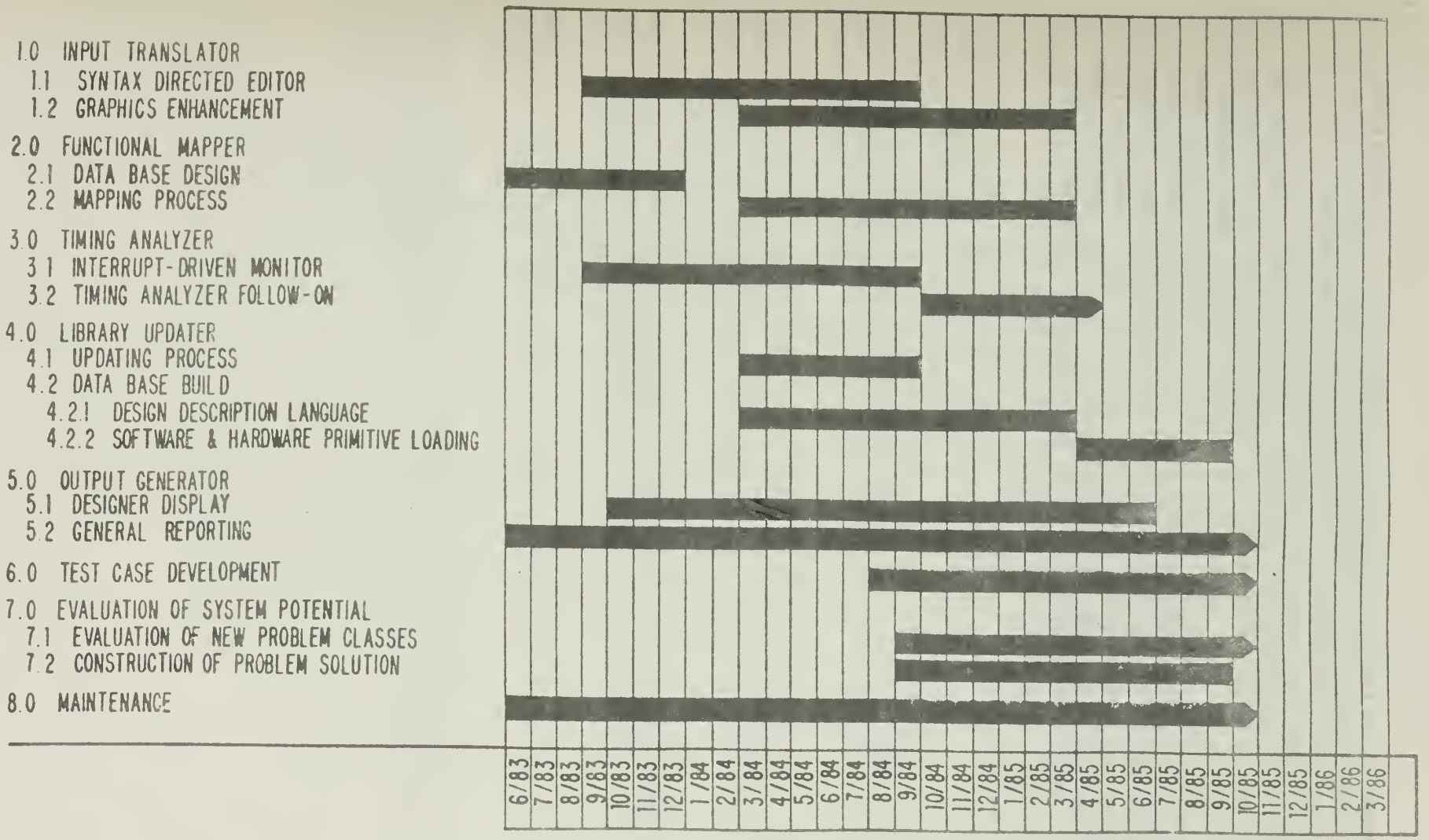
This task is an ongoing one which will be accomplished by the project staff, with assistance from the Computer Science Department staff. It includes keeping the software performing as designed, correcting problems, adjusting to changes required due to interface with the operating system and data base management system, and making required or desired enhancements. Enhancements or modifications required to implement a new class of problem are not included in this task but rather in Subtask 7.1.

C. Schedule

Figure 3 illustrates the project milestones. Although this is a working-level document, the milestones have been planned only to the subtask level. Since the thesis projects constitute systems development, it is appropriate that the thesis student perform the detailed planning and scheduling of the development. This will help him or her to learn the process and it will allow him or her to consider his or her level of participation for any quarter.

VII. Summary

The Computer Systems Design Environment Project is an effort to assist the computer systems designer by "talking his



COMPUTER SYSTEM DESIGN ENVIRONMENT DEVELOPMENT SCHEDULE

Figure 3

language" when interfacing with him, accurately generating the system which will satisfy his requirements, and providing a verification mechanism for validating the solution to his problem. The software for the Computer Systems Design Environment will be developed primarily by thesis students using accepted software engineering techniques with guidance from thesis advisors and assistance from staff personnel. The data collected during this project will be stored and accessed using a commercially available data base management system. The CSDE will be a prototype system which will initially address very specific problem classes and eventually as many classes as can be implemented. If the prototype answers the questions adequately, a production system will then be developed patterned from the prototype.

Bibliography
Computer System Design Environment

M. N. Matelan, "Automating the Design of Dedicated Real Time Control Systems," Preprint UCRL-78651 Lawrence Livermore Laboratory, August 21, 1976.

Alan A. Ross, "Feasibility Demonstration of an Automated Design System," SIGDA Newsletter, Vol 7, No. 1, March 1977.

Alan A. Ross, and Herschel H. Loomis, "Computer Aided Design of Microprocessor-Based Systems," Procreedings of the 15th Design Automation Conference, June 1978.

Alan A. Ross, "Computer Aided Design of Microprocessor-Based Controllers" PhD. Dissertation, University of California at Davis, 1978.

Hobart S. Cable, II, "Super-Cad: An Integrated Structure for Design Automation," M.S. Thesis, Air Force Institute of Technology, 1982.

Alan A. Ross, Herschel H. Loomis, and George G. Pollock, "Real Time Systems: An Approach to Computer-Aided Design of Hardware and Software," Presented at the 20th Annual Allerton Conference, October 7, 1982.

Barbara J. Sherlock, "User-Friendly, Syntax Directed Input To A Computer Aided Design System," M.S. Thesis, Naval Postgraduate School, June, 1983.

Martin R. Heilstedt, "Automated Design of Microprocessor-Based Digital Filters," Naval Postgraduate School, June, 1983.

Alan A. Ross, "Computer Systems Design Environment - Project Proposal," Presented to Office of Naval Research and Deputy Under Secretary of Defense for C3I for Evaluation, May, 1983.

APPENDIX A
GUIDE TO THE PREPARATION OF SOFTWARE DEVELOPMENT DOCUMENTATION

A. Guide to the Preparation of Software Development Documentation

The following paragraphs provide a guideline for the writing of each document required during the software development process. These documents will be written at appropriate times during the systems development cycle.

A.1. Requirements Document

The purpose of the Requirements Document is to define "what" the system should be able to do. The process used to obtain this information was described previously. The document requirements will follow that same analysis and design approach. Figure A-1 is an outline of the document contents and structure.

A.2. Program Specifications

The purpose of the Program Specification is to describe "HOW" the system requirements will be met. The specified data and logic required are described. Figure A-2 is an outline of the specification content and structure, and Figure A-3 is a sample form for documenting changes to existing programs.

A.3. Program Maintenance Manual

The purpose of the Maintenance Manual is to provide guidance to individuals who, in the future, may attempt to understand and/or modify the program. The amount of additional information required for this document is somewhat dependent on the level of detail of the program specification. The Maintenance Manual will be a combination of the Program Specification (current with the existing program), additional detailed logic (if required), test data, test results and a source listing. The student should place himself or herself in the position of someone unfamiliar with the program and try to detail the logic enough to provide a clear understanding of the program.

Comments should be provided liberally within the programs. They should be highlighted with asterisks (or equivalents) to make them noticeable. If the language has a compiler, a cross-reference list and object code should be included with the source. Test data necessary to thoroughly test the

program should be included along with corresponding test results.

A.4. User Documentation

The purpose of the User Documentation is to provide guidance for the end user of the program/system on its use. "HELP" screens will be used whenever practical so that user documentation can be reviewed online. A printed supplement will be written to provide guidance in areas where online access is not practical, e.g., program failures, system failures, logging on and running the program, and how to use help. Figure A-4 describes the contents and structure of the User Documentation. Since certain tasks lend themselves more to online user documentation than others, the split between what is going to be online and what will be in the supplementary document will be made by the thesis student. This split should be described in the "Purpose of the Document" Section of the Users' Manual.

REQUIREMENTS DOCUMENT OUTLINE

I. Purpose of the Document - This section discusses the significance of the Requirements Document in the development cycle. In the research environment, this document will be more technical in nature than the normal data processing development environment. The key point which must be made in this paragraph is that without a clear definition of "WHAT" the system needs to be able to do, the "WHAT" that is developed will not necessarily be the "WHAT" that is required.

II. Functional Decomposition - This section illustrates the functional breakdown of the module. Hierarchical block diagrams will be used to illustrate this breakdown. A numbering scheme will be used which relates a higher-level function to its lower-level subfunctions. Remember that a function at any level is exactly equivalent to the combination of its subfunctions.

III. Function Description - This section contains a one paragraph description of each function and subfunction. As the functions and subfunctions get smaller in scope, the descriptions will be more detailed.

IV. Description of Data - This section identifies and defines each category of data required by the module. All known data elements are identified for each category. A data dictionary should be generated containing this information.

V. Data-to-Function Relationship - This section describes the flow of data through functions. This illustration can be accomplished on one chart or a combination of several, one for each major process in the module. A narrative walk-through should accompany the charts.

VI. Transaction Identification - This section illustrates the transactions required to drive the module. It is recommended that proposed screen layouts be included.

VII. Summary - This section, as its name implies, summarizes the requirements of the system.

FIGURE A-1

PROGRAM SPECIFICATION OUTLINE

I. Purpose of the Document - This section discusses the significance of the Program Specification in the development cycle. It should explain that the design and logic of the program is contained in this document and that an expansion of this document will be the eventual maintenance documentation. The Program Specification says "HOW" we will do "WHAT" we documented in the Requirements Document.

II. Data Requirements - This section contains information on the input, work, and output data required by the program. As much as possible, this section should be generated from extracts of the data dictionary.

A. Input Data - This paragraph describes all input data including:

- o screen layouts
- o files
- o data base tables

B. Work Data - This paragraph describes all work data including:

- o temporary files
- o flags and indicators
- o arrays

C. Output Data - This paragraph describes all output data including:

- o screen layouts
- o report layouts
- o files
- o data base tables

III. Logic - This section contains an illustration and narrative describing the system logic. Pseudo-code should be used to construct the illustration. The narrative should be an overview of the logic, not a restatement of the pseudo-code and should clarify any complex or confusing issues.

IV. Environment - This section contains any special environmental considerations beyond those identified in Section VI paragraph B of the Software Development Plan. If there are none, this section will be left out.

V. Interface - This section contains a list of what other programs are called by this one and a second list of what other programs call this one. Any parameters to be passed are also identified.

VI. Language - This section identifies the language(s) in which the program will be written.

FIGURE A-2

PROGRAM MODIFICATION SPECIFICATION

NAME OF PROGRAM: _____

DATE: _____

NAME OF PERSON MAKING CHANGE: _____

CHANGES TO DATA REQUIREMENTS: _____

CHANGES TO LOGIC: _____

CHANGES TO INTERFACE: _____

MISCELLANEOUS CHANGES: _____

REQUIRED COMPLETION DATE: _____

REQUEST DATE: _____

REQUESTED BY: _____

FIGURE A-3

USER DOCUMENTATION

I. Purpose of the Document - This section discusses the significance of User Documentation. This section should describe how the split of user documentation breaks out, i.e., what is online and what is in the Users' Manual.

II. Description of Program/System - This section will describe the functions of the program/system. It will tell the end user what the system will do for him.

III. Input Requirements - This section will illustrate input required from the user. Printed layouts of input screens will be included and explained.

IV. Output Available - This section will illustrate output which is available from the program/system and will explain what is required to obtain it (Reference Section III).

V. Files - This section will describe the portion of the data base used by this program. Other files, outside the data base, will also be described.

VI. Errors - This section addresses responses to all predictable error situations.

FIGURE A-4

INITIAL DISTRIBUTION LIST

Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
Dudley Knox Library Code 0142 Naval Postgraduate School Monterey, CA 93943	2
Office of Research Administration Code 012A Naval Postgraduate School Monterey, CA 93943	1
Chairman, Code 52Hq Department of Computer Science Naval Postgraduate School Monterey, CA 93943	40
Professor Alan A. Ross, Code 52Rs Department of Computer Science Naval Postgraduate School Monterey, CA 93943	15